Solve logical problems using python

1. **Area**

```
def calculate_area():
    print("Choose the shape to calculate the area:")
    print("1. Rectangle")
    print("2. Square")
    print("3. Triangle")
    print("4. Circle")

    choice = input("Enter the number of your choice: ")

    if choice == '1':
        length = float(input("Enter the length of the rectangle: "))
        width = float(input("Enter the width of the rectangle: "))
        area = length * width
        print(f"The area of the rectangle is {area}")

    elif choice == '2':
        side = float(input("Enter the side length of the square: "))
        area = side ** 2
        print(f"The area of the square is {area}")

    elif choice == '3':
        base = float(input("Enter the base of the triangle: "))
        height = float(input("Enter the height of the triangle: "))
        area = 0.5 * base * height
        print(f"The area of the triangle is {area}")

    elif choice == '4':
        radius = float(input("Enter the radius of the circle: "))
        area = 3.14159 * radius ** 2
        print(f"The area of the circle is {area}")
```

```
    else:
        print("Invalid choice. Please choose a valid shape.")

calculate_area()
```

2. **Reversestring**

```
def reverse_string():
    # Prompt the user to enter a string
    user_string = input("Enter a string to reverse: ")

    # Reverse the string using slicing
    reversed_string = user_string[::-1]

    # Print the reversed string
    print(f"The reversed string is: {reversed_string}")

# Run the function to reverse the string
reverse_string()
```

3. **Largest element**

```
def find_largest_element():
    # Prompt the user to enter a list of numbers separated by spaces
    numbers = input("Enter numbers separated by spaces: ").split()

    # Convert the input strings to integers
    numbers = [int(num) for num in numbers]

    # Find the largest number in the list
    largest = max(numbers)

    # Print the largest number
```

```
    print(f"The largest element is: {largest}")

# Run the function to find the largest element
find_largest_element()
```

4. Sum of elements

```
def find_sum_of_elements():
    # Prompt the user to enter a list of numbers separated by spaces
    numbers = input("Enter numbers separated by spaces: ").split()

    # Convert the input strings to integers
    numbers = [int(num) for num in numbers]

    # Calculate the sum of the numbers in the list
    total_sum = sum(numbers)

    # Print the sum of the numbers
    print(f"The sum of the elements is: {total_sum}")

# Run the function to find the sum of elements
find_sum_of_elements()
```

5. Duplicate

```
def find_duplicate_elements():
    # Prompt the user to enter a list of numbers separated by spaces
    numbers = input("Enter numbers separated by spaces: ").split()

    # Convert the input strings to integers
    numbers = [int(num) for num in numbers]

    # Create a set to track seen numbers and a list to store duplicates
```

```
    seen = set()
    duplicates = []

    # Iterate through the numbers and identify duplicates
    for num in numbers:
        if num in seen:
            if num not in duplicates:  # Avoid adding the same duplicate multiple
times
                duplicates.append(num)
        else:
            seen.add(num)

    # Print the duplicates
    if duplicates:
        print(f"The duplicate elements are: {duplicates}")
    else:
        print("There are no duplicate elements.")

# Run the function to find duplicate elements
find_duplicate_elements()
```

6. List is empty

```
def check_if_list_is_empty():
    # Prompt the user to enter a list of elements separated by spaces
    elements = input("Enter elements separated by spaces (leave empty for
an empty list): ").split()

    # Check if the list is empty
    if not elements:
        print("The list is empty.")
    else:
        print("The list is not empty.")
```

```
# Run the function to check if the list is empty
check_if_list_is_empty()
```

7. Programs on dictionaries

Creating and accessing dictionaries

```
def create_and_access_dictionary():
    # Creating a dictionary
    student = {
        'name': 'John Doe',
        'age': 25,
        'grade': 'A',
        'courses': ['Math', 'Science', 'History']
    }

    # Accessing elements in the dictionary
    print(f"Name: {student['name']}")
    print(f"Age: {student['age']}")
    print(f"Grade: {student['grade']}")
    print(f"Courses: {', '.join(student['courses'])}")

# Run the function to create and access a dictionary
create_and_access_dictionary()
```

Updating dictionary

```
def update_dictionary():
    # Creating a dictionary
    student = {
        'name': 'John Doe',
        'age': 25,
```

```
        'grade': 'A',
        'courses': ['Math', 'Science', 'History']
    }

    # Updating a value in the dictionary
    student['age'] = 26
    student['courses'].append('English')

    # Printing the updated dictionary
    print("Updated Student Dictionary:")
    for key, value in student.items():
        if isinstance(value, list):
            value = ', '.join(value)
        print(f"{key}: {value}")

# Run the function to update a dictionary
update_dictionary()
```

Merging dictionary

```
def merge_dictionaries():
    # Creating two dictionaries
    dict1 = {'a': 1, 'b': 2}
    dict2 = {'c': 3, 'd': 4}

    # Merging dictionaries using update() method
    merged_dict = dict1.copy()
    merged_dict.update(dict2)

    # Printing the merged dictionary
    print("Merged Dictionary:", merged_dict)

# Run the function to merge dictionaries
```

```
merge_dictionaries()
```

Finding keys and values

```
def find_keys_and_values():
    # Creating a dictionary
    student = {
        'name': 'John Doe',
        'age': 25,
        'grade': 'A',
        'courses': ['Math', 'Science', 'History']
    }

    # Finding keys and values in the dictionary
    print("Keys in Student Dictionary:")
    for key in student.keys():
        print(key)

    print("\nValues in Student Dictionary:")
    for value in student.values():
        if isinstance(value, list):
            value = ', '.join(value)
        print(value)

# Run the function to find keys and values in a dictionary
find_keys_and_values()
```